Juha Harju

# Konenäkö-opetussovellus

Opinnäytetyö

Kevät 2012

Tekniikan yksikkö

Tietotekniikan koulutusohjelma

Mekatroniikan suuntautumisvaihtoehto

Seinäjoen ammattikorkeakoulu
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

SEINÄJOEN AMMATTIKORKEAKOULU

OPINNÄYTETYÖN TIIVISTELMÄ

Koulutusyksikkö: Tekniikan yksikkö
Koulutusohjelma: Tietotekniikka
Suuntautumisvaihtoehto: Mekatroniikka

Tekijä: Juha Harju

Työn nimi: Konenäkö-opetussovellus

Ohjaaja: Martti Lehtonen

Vuosi: 2012 Sivumäärä: 42 Liitteiden lukumäärä: 3

---

Työn tarkoituksena oli luoda Motoman-robotille ja DVT SmartImage Sensor -konenäkökameralle opetussovellus, jolla voisi esitellä robotin toimintaa konenäkökameran ohjauksella. Työtä voisi käyttää hyväksi opetustunneilla esitellessä teollisuusrobotin toimintaa ja lisäksi messuilla, joilla robotti on ollut mukana pienen kokonsa ja helpon siirrettävyyden vuoksi.

Työssä täytyi ensiksi saada laitteet kommunikoimaan koulun lähiverkon kautta. Laitteet olivat aiemmin olleet yhteydessä, mutta yhteysasetukset eivät olleet enää kunnossa. Yhteyden palauttamisessa ilmeni kuitenkin ongelmia ja työssä täytyi perehtyä huolellisesti myös laitteiden manuaaleihin vikojen ratkaisemiseksi.

Luodussa sovelluksessa Motoman-robotti liikuttelee työkappaletta, jonka paikkatiedot saadaan konenäkökameran ottamasta kuvasta. Kameralta saatujen paikkatietojen avulla robotti osaa automaattisesti hakea kappaleen työalueelta.

Työstä on toivottavasti apua seuraaville käyttäjille, jotka haluavat luoda vastaavanlaisen sovelluksen Motoman-robottia ja DVT-konenäkökameraa käyttäen.

Asiasanat: konenäkö, robotiikka, Motoman, DVT

SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

**Thesis abstract**

Faculty: School of Technology
Degree programme: Information Technology
Specialisation: Mechatronics

Author: Juha Harju

Title of the thesis: Machine vision teaching application

Supervisor: Martti Lehtonen

Year: 2012 Number of pages: 42 Number of appendices: 3

The aim of the thesis was to create a teaching application for a Motoman robot and a DVT SmartImage Sensor machine vision camera that could be used to demonstrate the robot's operations with machine vision guidance. The application could be used in lectures to show how an industrial robot works and also at trade fairs, where the robot has been on display due to its small frame and easy portability.

First, it was necessary to make the devices communicate with each other through the school's local area network. The devices had been communicating before but the connection settings were not working anymore. However, there were some problems with restoring the connection between the devices and to solve that problem the manuals of the devices had to be revised.

In the created application, the Motoman robot is moving an object whose positional information is acquired from the image taken with the machine vision camera. With the position information received from the camera, the robot can automatically pick up the object from a workspace.

The application may be of help for the successors using the Motoman robot and DVT machine vision camera to create similar applications.

Keywords: machine vision, robotics, Motoman, DVT

# SISÄLLYS

**TIIVISTELMÄ**

**ABSTRACT**

**SISÄLLYS**

**1 JOHDANTO ..... 6**

**2 TEOLLISUUSROBOTIT ..... 7**

2.1 Yleistä teollisuusroboteista ..... 7

2.2 Robottien mekaaninen rakenne ..... 7

2.3 Vakiintuneita robottityyppejä ..... 8

2.4 Teollisuusrobottien käyttökohteet ..... 9

**3 KONENÄKÖ ..... 11**

3.1 Mitä on konenäkö? ..... 11

3.2 Käyttösovellukset ..... 11

3.3 Optiikka ..... 12

3.4 Valaistus ..... 12

**4 LAITTEISTO JA OHJELMISTOT ..... 14**

4.1 DVT SmartImage Sensor 542C ..... 14

4.2 Motoman SV3X -teollisuusrobotti ..... 15

4.3 Moxa NPort 5230 -sarjaporttipalvelin ..... 16

4.4 PC ja FrameWork-ohjelmisto ..... 17

**5 ASETUKSET ..... 18**

5.1 Kameran asetukset ..... 18

5.2 Moxa NPort 5230:n asetukset ..... 18

5.3 Motoman XRC:n asetukset ..... 21

**6 DVT FRAMEWORK -OHJELMISTON KÄYTTÄMINEN ..... 23**

6.1 SoftSensors ..... 23

6.2 Kappaleen paikoitus ..... 24

6.3 Skriptien käyttö ..... 24

6.3.1 Foreground Script ..... 26

6.3.2 Background Script....27
6.4 Kalibrointi robotin käyttämään koordinaatistoon....28
**7 ROBOTIN OHJELMA....30**
**8 KONENÄKÖSOVELLUKSEN LUOMINEN....31**
**9 YHTEENVETO....36**
**LÄHTEET....37**
**LIITTEET....38**
LIITE 1: Robotin ohjelma....38
LIITE 2: ForeGround Script....40
LIITE 3: BackGround Script....41

# 1 JOHDANTO

Konenäkö on erittäin monipuolinen anturi, joka muistuttaa näköaistia. Ohjelmoitavuutensa ansiosta se soveltuu moniin erilaisiin sovelluksiin. Konenäköjärjestelmä koostuu valonlähteestä, kohteesta, kamerasta, tietokoneesta ja siinä toimivasta kuvankäsittelyohjelmasta, joka tulkitsee kuvan automaattisesti. Konenäön käyttö teollisuudessa on yleistynyt huomattavasti tietotekniikan kehittymisen ja sen mukana tulleen parantuneen suorituskyvyn myötä.

Opinnäytetyön tarkoitus oli saada koulun DVT-merkkinen konenäkökamera kommunikoimaan Motoman-robotin kanssa ja luoda opetussovellus, jolla voisi esitellä robotin toimintaa konenäön ohjauksella.

Työssä kerrotaan laitteista, niiden liittämisestä, asetusten määrittelystä, aiheeseen liittyvistä ohjelmista sekä laitteiden ohjelmoinnista.

# 2 TEOLLISUUSROBOTIT

## 2.1 Yleistä teollisuusroboteista

Teollisuusrobotti on kansainvälisen robottiyhdistyksen mukaan uudelleen ohjelmoitavissa oleva monipuolinen vähintään kolminivelinen mekaaninen laite, joka on suunniteltu liikuttamaan kappaleita, osia, työkaluja tai erikoislaitteita ohjelmoitavin liikkein monenlaisten tehtävien suorittamiseksi teollisuuden sovelluksissa. (Kuivanen 1999, 12-13.)

Tähän mennessä teollisuusrobotteja valmistaneita yrityksiä on ollut yli 500 ja jokaisen valikoimiin on kuulunut useita robottimalleja. Erilaisia robottimalleja on suunniteltu useita tuhansia patenttien ja eri sovellusten vuoksi. Tuotannossa on kuitenkin muutamia teollisuusrobottimalleja, jotka pystytään ohjelmoimaan suorittamaan useita erilaisia sovelluksia, ja joita valmistetaan vuodessa tuhansittain. (Kuivanen 1999, 12-13.)

Standardi ISO 8373 määrittelee teollisuusrobottien sanastoa ja myös yleisimmät robottimallit mekaanisen rakenteen mukaan. (Kuivanen 1999, 12-13.)

## 2.2 Robottien mekaaninen rakenne

Robotti koostuu tukivarsista, joista kaksi liikkuu toistensa suhteen joko tietyn suoran suuntaan tai sen ympäri. Tätä käsitteellistä akselia kutsutaan usein niveleksi. (Kuivanen 1999, 15-16.)

Nivelten avulla tukivarret muuttavat keskinäisiä asentoja ja asemiaan. Yhtä robotin perusliikettä eli niveltä sanotaan vapausasteeksi. Vapausastetta kohti on yleensä yksi toimilaite kuten moottori tai sylinteri. Kiertyvänivelisillä roboteilla on yleensä

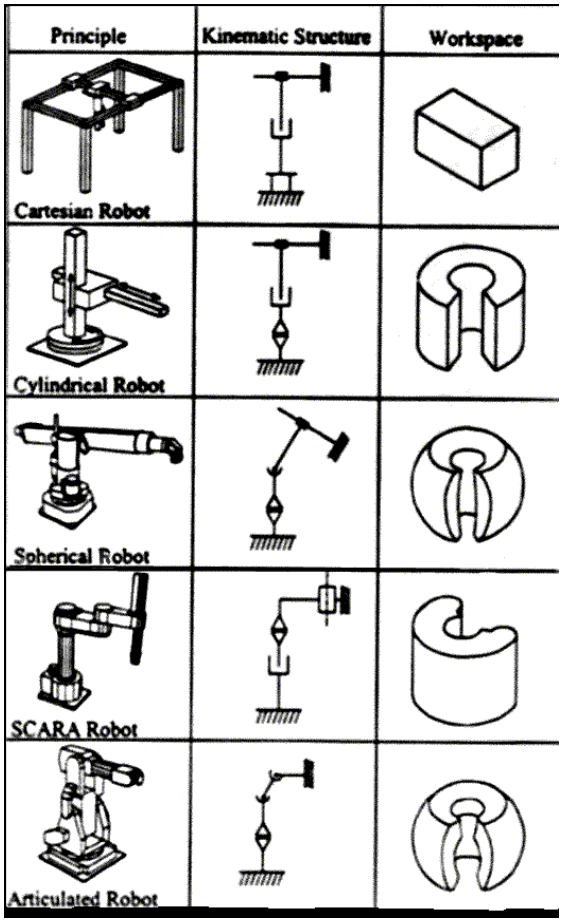
kuusi vapausastetta, jotka mahdollistavat robotin soveltuvuuden monenlaisiin tehtäviin. (Kuivanen 1999, 15-16.)

Kytkemällä mekaanisia vapausasteita eri tavalla yhteen ja muuttamalla vapausasteiden liikematkoja saadaan lukuisia erilaisia robotteja. (Kuivanen 1999, 15-16.)

## 2.3 Vakiintuneita robottityyppejä

Yleisimpiä robottityyppejä ovat

- suorakulmaiset robotit
- Scara-robotit
- kiertyväniveliset robotit ja
- sylinterirobotit. (Kuivanen 1999, 16-17.)




KUVIO 1. Robottityypit ja niiden mekaaninen rakenne. (RobotWorks, [Viitattu 26.3.2012].)

Suorakulmaisten robottien kolme ensimmäistä vapausastetta ovat lineaarisia. Tyypillinen edustaja on portaalirobotti, jonka rakenne on tuettu työalueen nurkista palkeilla. (Kuivanen 1999, 16-17.)

Scara-robotti muistuttaa vaakatasossa liikkuvaa käsivartta. Kolmella kiertyvällä nivelellä työkalu saadaan vaakatasolla oikeaan kohtaan ja kiertymäkulmaan. Neljäs vapausaste on lineaarinen pystyliike. (Kuivanen 1999, 16-17.)

Kiertyvänivelinen on tavallisin teollisuusrobottityyppi, jossa kaikki vapausasteet ovat kiertyviä. (Kuivanen 1999, 16-17.)

Sylinterirobotin nimitys on peräisin sylinterikoordinaatistosta. (Kuivanen 1999, 16-17.)

## 2.4 Teollisuusrobottien käyttökohteet

Jotkin teollisuuden työtehtävät suosivat robottien käyttämistä ihmistyövoiman sijaan. Tällaisia tapauksia ovat esimerkiksi:

- Ihmisille haitallinen työ. Työn ja työympäristön ollessa vaarallinen, epäterveellinen tai muuten vaan epämukava, on syytä harkita teollisuusrobottia suorittamaan tehtävä. Teollisuudessa ihmisille haitallisia työtapauksia on monenlaisia, kuten painevalu, ruiskumaalaus, kaarihitsaus ja pistehitsaus. (Groover 2008, 225-226.)

- Toistuva työkierto. Robotti pystyy suorittamaan liikkeiltään yksinkertaisia ja itseään toistavia työtehtäviä paremmin kuin ihminen. Varsinkin pitkällä aikavälillä ihminen helposti kyllästyy suorittamaan koko ajan vain samaa työtehtävää, jolloin työn laatu ja toistojen nopeus heikkenevät. Teollisuusrobotilla ei vastaavaa tapahdu, vaan työn laatu ja nopeus pysyvät muuttumattomina. (Groover 2008, 225-226.)

- Vaikea käsiteltävyys. Jotkin teollisuusrobotit soveltuvat hyvin tehtäviin, jotka sisältävät ihmiselle liian raskaita tai vaikeasti käsiteltäviä kappaleita tai työkaluja. (Groover 2008, 225-226.)

- Usean työvuoron tehtävät. Robotti maksaa itsensä nopeammin takaisin, jos se sijoitetaan työtehtävään jota suoritetaan yhden työvuoron sijaan useassa työvuorossa. Tällöin robotti ei korvaa pelkästään yhtä työntekijää, vaan kaksi tai kolme työntekijää. (Groover 2008, 225-226.)

Robotteja käytetään todella laajasti teollisuuden käyttösovelluksissa ja suurin osa näistä sovelluksista liittyy tuotantoon. Teollisuusrobotin käyttösovelluksia ovat: materiaalin käsittely, työstö toiminnot ja kokoaminen sekä tarkastaminen. (Groover 2008, 225-226.)

# 3 KONENÄKÖ

## 3.1 Mitä on konenäkö?

Konenäkö on elektroninen vaihtoehto ihmisnäölle ja manuaaliselle tuotteiden tarkastamiselle, joka auttaa yrityksiä kasvattamaan tuottavuutta ja säästämään rahaa havaitsemalla epätäydelliset tuotteet 100 prosentin tarkkuudella (DVT 2004).

Konenäkö tarkoittaa koneelle luotua näkökykyä, jota voidaan käyttää esimerkiksi etsimään tuotteesta epäkohtia tunnistamalla onko jokin osa paikalla vai puuttuuko se, tekemään mittauksia, lukemaan koodeja ja tunnistamaan värejä. Toiminta voi aluksi näyttää monimutkaiselta, mutta koneen silmänä toimiva kamera etsii kaksiulotteiselta tasolta pääosin vain muutoksia kontrastissa. Koneen ohjelman ja käyttäjän määrittelemien sääntöjen mukaan määräytyy kuinka kamera arvioi muutokset kontrastissa ja kuinka saatu tieto käsitellään. Kunnollinen valaistus on myös tärkeää konenäköjärjestelmää käytettäessä, koska kameran toiminnot ovat riippuvaisia kontrastin muutoksista. (Cognex 2003.)

## 3.2 Käyttösovellukset

Konenäköä käytetään yleisesti robottijärjestelmissä ja sitä tarvitaan silloin, kun perinteinen anturointi ei enää riitä ja kun halutaan minimoida mekaanisten paikoittimien tai kiinnittimien tarve. Sovelluskohteet ovat lähinnä erikoissovelluksia, koska konenäkö ei kaikissa tapauksissa pysty hintansa puolesta kilpailemaan yksinkertaisempien mekaanisten tai anturipohjaisten ratkaisujen kanssa. (Kuivanen 1999, 56.)

Karkeasti näköjärjestelmän tehtävät robottisovelluksissa voidaan jakaa kolmeen ryhmään:

- Kappaleen tai kohteen sijainnin määrittäminen.
- Luokittelu eli kohteen tunnistus tai luokittelu laadun, muodon, värin, tunnisteiden, koon tai kohteessa sijaitsevan tekstin perusteella.
- Kohteen mittaus robotin liikeohjelman muokkaamiseksi tai luomiseksi. (Kuivanen 1999, 56.)

### 3.3 Optiikka

Konenäköoptiikassa käytetään yleensä tavallisia valokuvauskameroista tuttuja objektiiveja. Linssin valinnassa täytyy ottaa huomioon asioita, riippuen halutusta kuvanlaadusta, kuva-alasta, resoluutiosta ja kohteen etäisyydestä. Optiikasta käytetään myös termejä fokusointi eli tarkennus ja kameran aukko, jolla säädetään tulevaa valon määrää. (Cognex 2003.)

### 3.4 Valaistus

Yksi tärkeimmistä osista konenäköjärjestelmää on kuvauspaikan ja kuvattavan kappaleen valaistus. Kuvausalueen täytyy olla hyvin valaistu ja valaistuksen täytyy pysyä tasaisena, koska vaihtelut voivat aiheuttaa virheitä kameran suorittaman tarkastelun tuloksissa. Tästä syystä yleensä ei luoteta alueen normaaliin valaistukseen tai luonnonvaloon, vaan laitteistolle on oma siihen kohdistettu erikoisvalaistus. (Groover 2008, 685.)

Valonlähteinä käytetään hehkulamppuja, loisteputkilamppuja, matala- ja suurpainenatriumlamppuja, lasereita ja LED-valaistusta. Valaistusteknologioiden lisäksi valonlaatuun vaikuttaa paljon mistä suunnasta ja missä kulmassa valo tulee kohteeseen. (Groover 2008, 685.)

Etuvalaistuksessa valonlähde sijaitsee kuvattavaan kappaleeseen nähden samalla puolella kuin kamera. Tämä tuottaa kohteesta heijastunutta valoa, joka mahdollistaa hyvin pinnan muotojen ja piirteiden tarkastelun. Takavalaistuksessa valonlähde sijaitsee kamerasta katsottuna kohteen takana. Tämä muodostaa tumman var-

jokuvan kohteesta, joka erottuu hyvin valoisasta taustasta. Takavalaistusta käytetään määritettäessä kohteen mittoja ja erottamaan eri osat ja ulkolinjat kuvasta. Sivuvalaistus aiheuttaa varjoja, joiden avulla voi huomata epäsäännöllisyyksiä muuten tasaisella pinnalla. Tätä voidaan käyttää epäkohtien ja virheiden etsimisessä kohteen pinnasta. (Groover 2008, 685.)

Jäsentynyt valaistus tarkoittaa jonkin erikoisvalokuvion suuntaamista kohteeseen. Täten voidaan tehostaa joitain tiettyjä geometrisia yksityiskohtia. (Groover 2008, 685.)

Yksi vaihtoehto on valaista alue vilkkuvalla korkean intensiteetin valolla. Tämä auttaa liikkuvien kohteiden kuvauksessa, koska kohde näyttää pysyvän paikallaan valon välähdellessä. (Groover 2008, 685.)

# 4 LAITTEISTO JA OHJELMISTOT

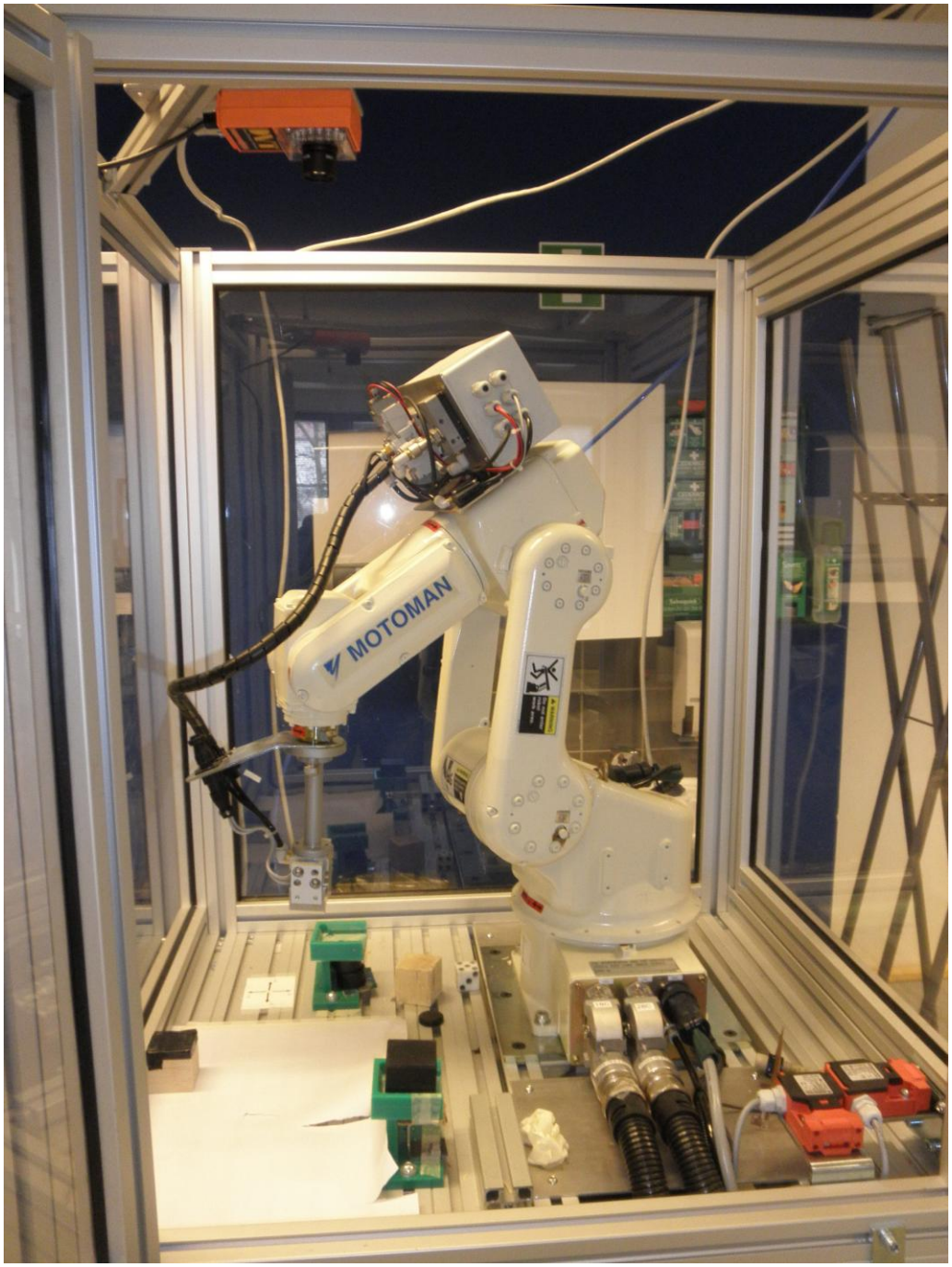
## 4.1 DVT SmartImage Sensor 542C

DVT SmartImage Sensor 542C -kamerassa on 640x480 resoluution tarkkuudella toimiva CCD-värikenno, joka tekee kamerasta hyvän vaihtoehdon moniin värien tarkastelua vaativiin sovelluksiin. (Cognex 2003).

SmartImage Sensor 500 -sarjan kameroiden mukana tulee DVT FrameWork -ohjelmisto, jolla voi säätää SmartImage Sensorin asetuksia sekä määritellä ja ohjelmoida kameran kuvista antamat tiedot käyttäjän tarpeiden mukaan. (Cognex 2003.)


KUVIO 2. DVT SmartImage Sensor, Legend 542C.

## 4.2 Motoman SV3X -teollisuusrobotti




KUVIO 3. Motoman SV3X -teollisuusrobotti.

Motoman SV3X -robotin ominaisuuksiin kuuluu 677 mm:n ulottuvuus, kuusi akselia, 3 kg:n hyötykuorma ja laajin työsäde luokassaan. SV3X:n käyttösovelluksiin kuuluvat materiaalin käsittely, kokoaminen, pakkaus, konepalvelu ja pistehitsaus. (RobotWorks, [Viitattu 26.3.2012].)

Robotti on koululla enimmäkseen opetuskäytössä, mutta pienen koon ja helpon siirrettävyyden ansiosta se on ollut esillä messuilla koulua mainostettaessa.

### 4.3 Moxa NPort 5230 -sarjaporttipalvelin

Moxa NPort 5200 -sarjan sarjaporttipalvelin on luotu teollisten sarjaliikennelaitteiden kytkemiseksi lähiverkkoon tai Internetiin. Nport-adapterin avulla laitteen kanssa voidaan kommunikoida lähiverkko- tai internetyhteyden välityksellä IP-osoitetta käyttäen. (Moxa 2011.)

Moxa Nport 5230:ssa on RS-232- ja RS-422/485-liitäntämahdollisuudet ruuviliittimin ja RJ-45-pistoke Ethernet-liitännälle. Laitteen asetuksiin pääsi käsiksi verkkoselaimen avulla.

Sarjaporttipalvelinta tarvittiin yhdistämään Motoman-robotti koulun lähiverkkoon, koska yhteys konenäkökameran kanssa robotin Ethernet-liitännän kautta ei toiminut.


KUVIO 4. Moxa NPort 5230.

### 4.4 PC ja FrameWork-ohjelmisto

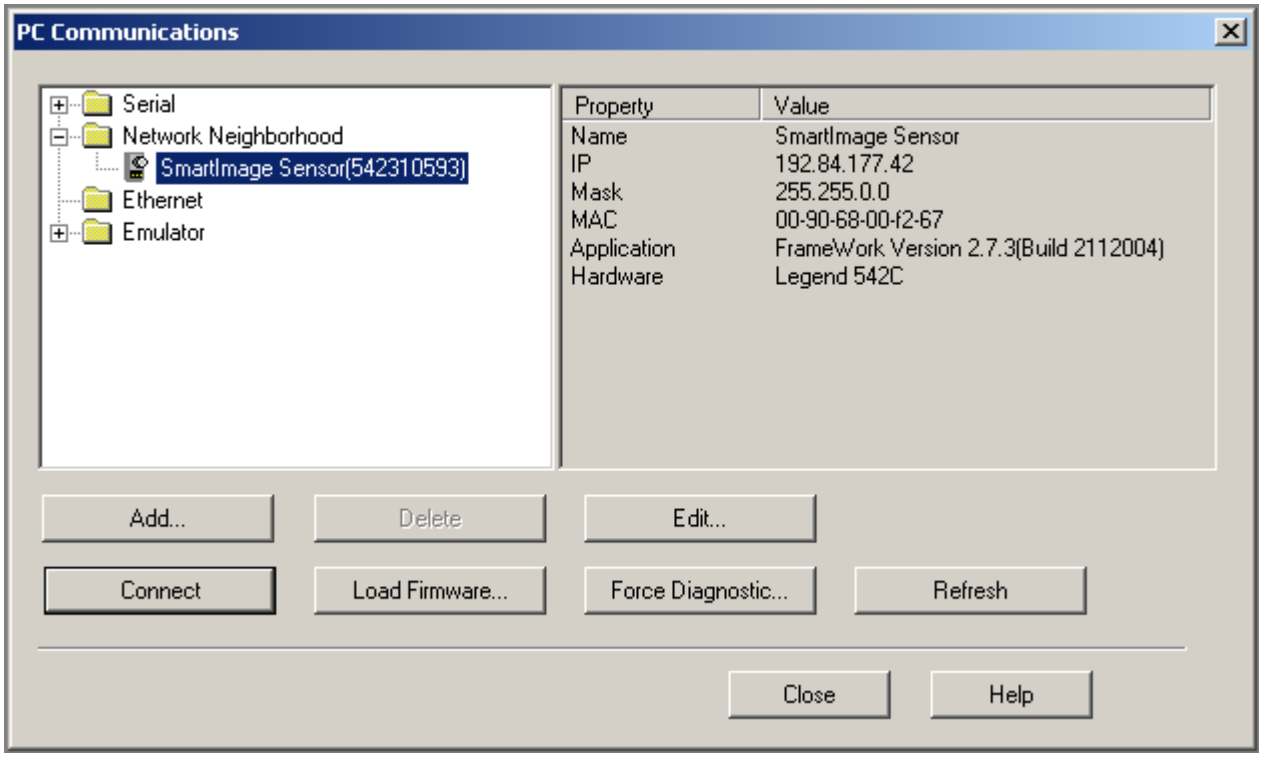
Tietokoneena toimi Dell Optiplex 760 ja sitä käytettiin kameran ohjelmoimiseen sekä yhteysasetusten muuttamiseen. Kamera pystyi toimimaan itsenäisesti ilman tietokonetta, mutta muutokset asetuksiin ja ohjelmaan täytyi tehdä tietokoneen avulla.

Aluksi tietokoneelle asennettiin DVT-kameroille tarkoitettu uusin DVT Intellect -ohjelmisto. Sen ohjelmointipuolelta kuitenkin puuttui osa Motomanin komennoista, joita haluttiin käyttää. Näin päädyttiin FrameWork-ohjelmaan, jonka sai ladattua ilmaiseksi kameran valmistajan kotisivuilta.

# 5 ASETUKSET

## 5.1 Kameran asetukset

Kameran IP-osoite ja aliverkon peite täytyi asettaa samalle alueelle PC:n osoitteiden kanssa, että yhteyden luominen onnistui. Lisäksi kameraan täytyi ladata firmware-versio, joka oli yhteensopiva Framework version 2.7.3 kanssa.

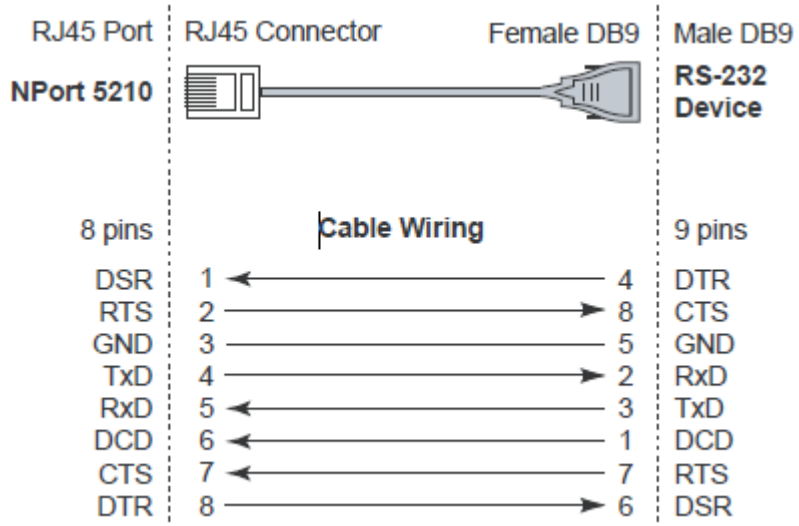



KUVIO 5. Kameran verkkoasetukset.

## 5.2 Moxa NPort 5230:n asetukset

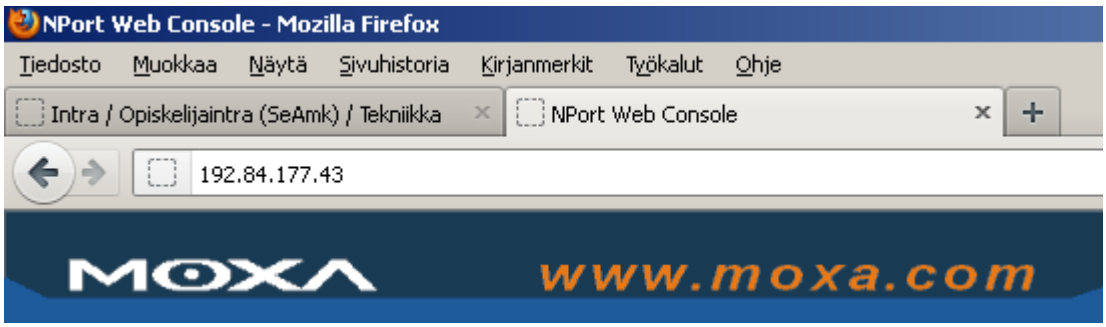
Moxa Nport 5230 kytkettiin robottiin muuten kuvion 6 mukaisesti, mutta kaapelin RJ45-pää oli purettu ja tarvittavat johdot liitettiin Moxan RS-232-portin ruuviliitti-

miin. Tässä tapauksessa täytyi kytkeä RxD tulevalle liikenteelle ja TxD lähtevälle liikenteelle, sekä lisäksi signaalinmaadoitus GND.



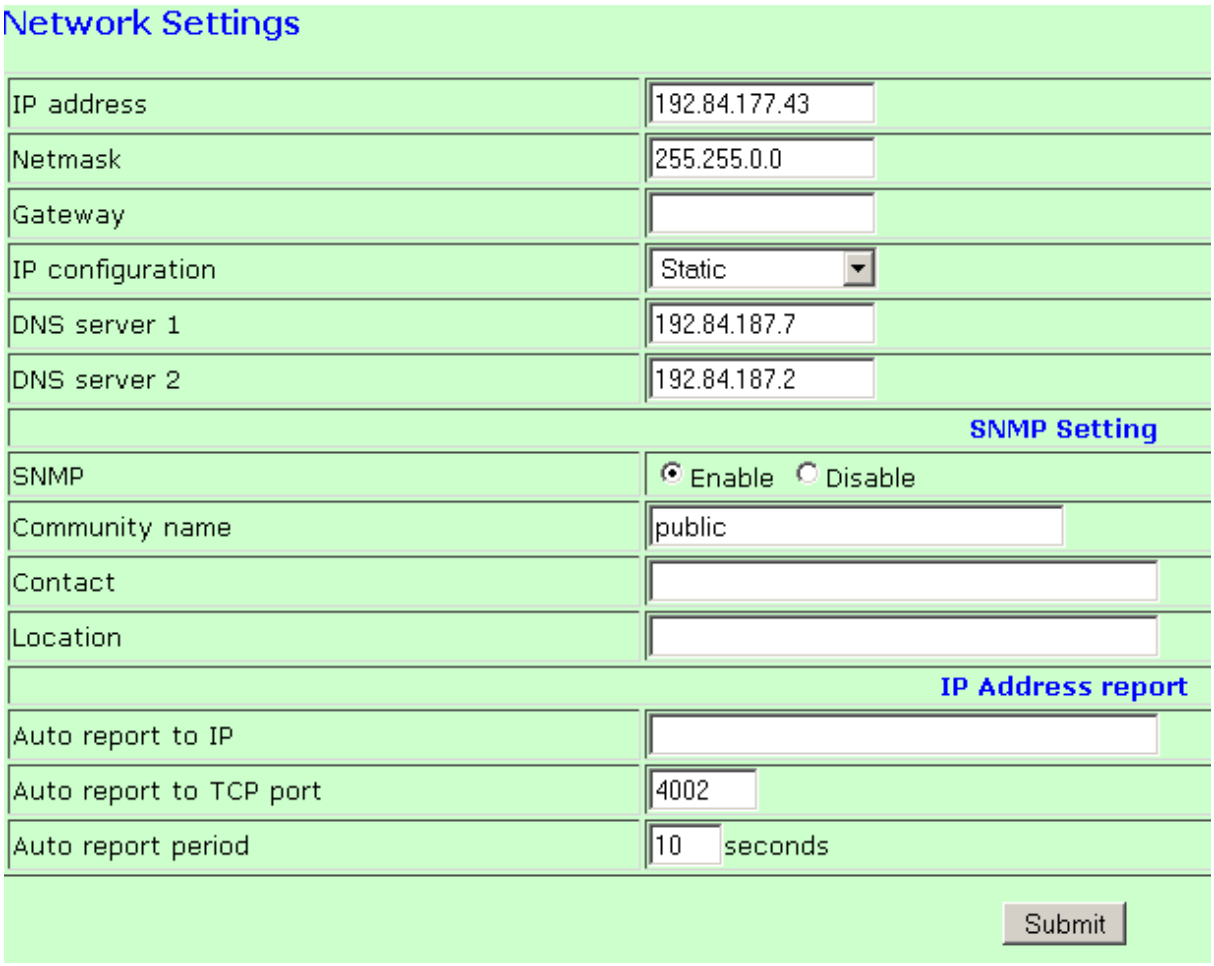

KUVIO 6. Robotin ja NPort 5230:n välinen kytkentä. (Moxa 2011.)

NPort täytyi aluksi palauttaa takaisin tehdasasetuksiin, koska sen käyttämä IP-osoite ei ollut tiedossa. Myös tietokoneen IP-osoite ja aliverkon peite täytyi muuttaa sarjaporttipalvelimen kanssa samalle alueelle. Tämän jälkeen Moxaan saatiin yhteys oletus-IP-osoitetta 192.168.127.254 käyttäen, ja asetuksiin päästiin käsiksi syöttämällä kyseinen IP-osoite verkkoselaimen osoiteriville.

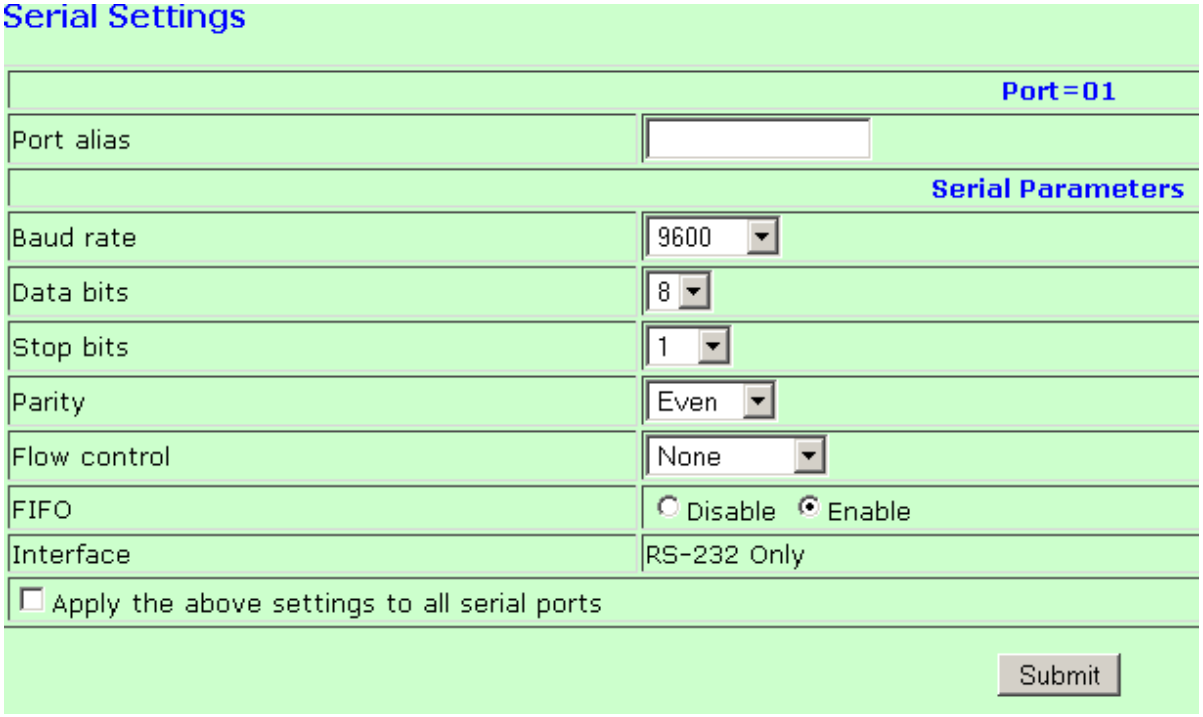



KUVIO 7. Yhteyden muodostaminen Moxa NPort 5230:aan.

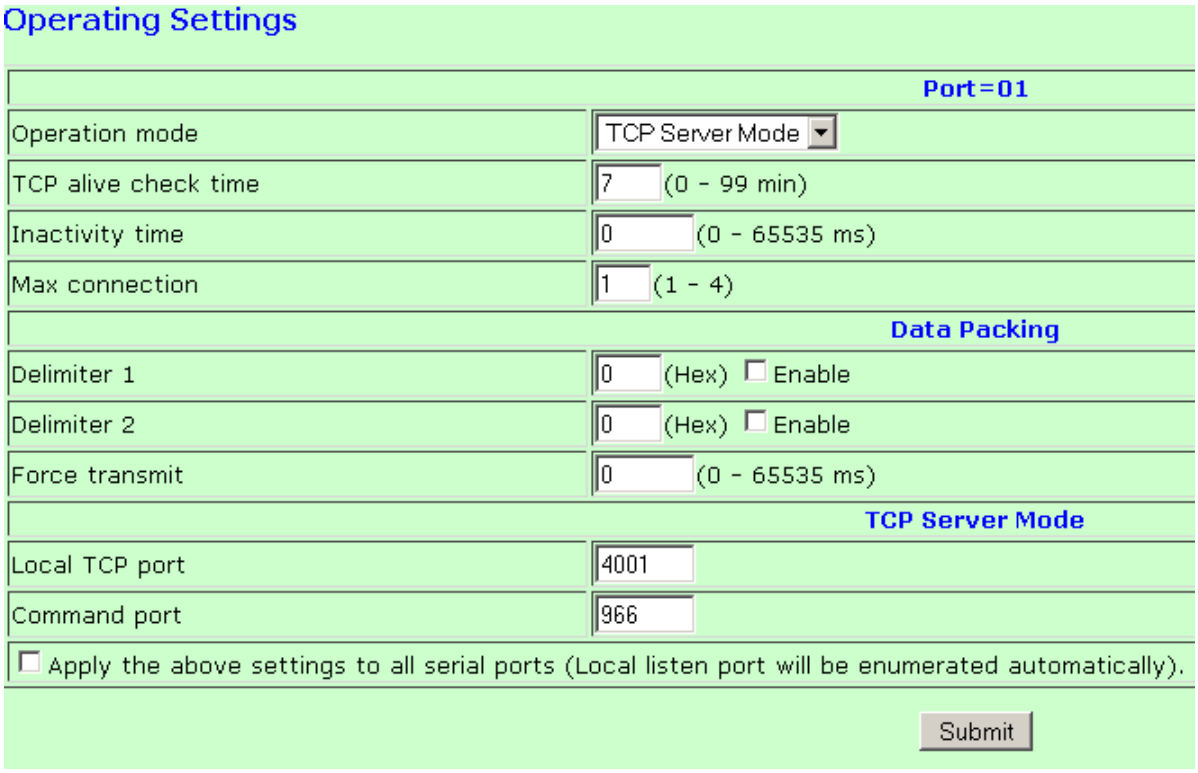
Moxan asetukset määriteltiin kuvioiden 8, 9 ja 10 mukaisesti.




KUVIO 8. Moxa NPort 5230:n verkkoasetukset.




KUVIO 9. Sarjaportin asetukset.




KUVIO 10. Toimintatavan ja TCP-portin määrittäminen.

### 5.3 Motoman XRC:n asetukset

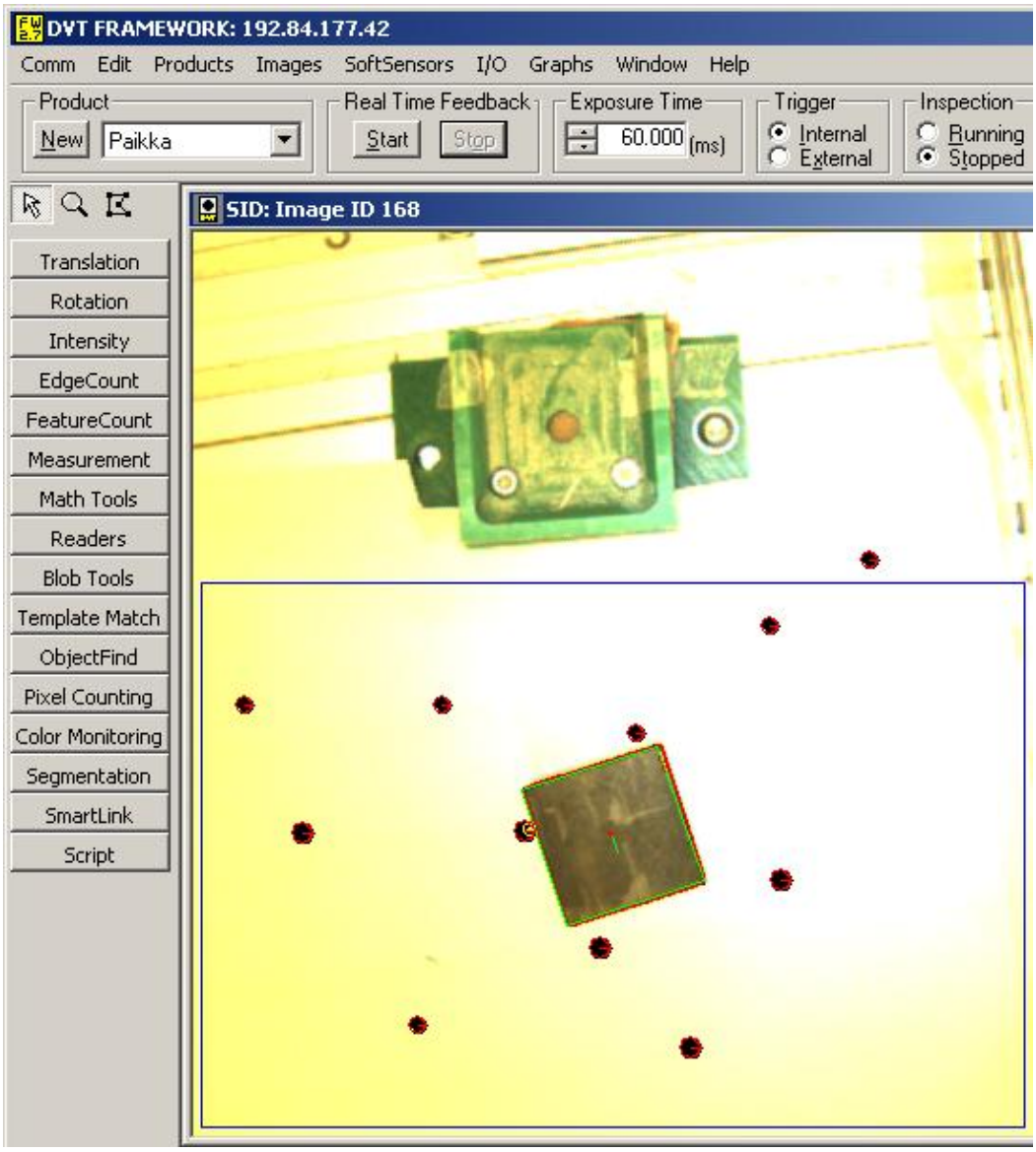
Robotin sarjaportin kommunikointiasetusten täytyi olla samat kuin Moxa Nport 5230:lla, jotta yhteys laitteiden välillä toimi. Asetuksia pääsi muuttamaan robottiohjaimen Parametrit-valikosta, mutta muutoksien tekeminen ei ollut tarpeellista, koska asetukset olivat jo valmiiksi oikeissa arvoissa.

TAULUKKO 1. Motoman XRC:n sarjaportin asetukset.
(DVT-Motoman Driver, [Viitattu 2.4.2012].)

| **Parametrit** | **Selitys** |
|---|---|
| RS000 = 2 | BSC LIKE |
| RS030 = 8 | 8 data bits |
| RS031 = 0 | 1 stop bits |
| RS032 = 2 | Even parity |
| RS033 = 7 | 9600 baud |

XRC:n täytyi olla Local mode -tilassa, että kommunikointi DVT-kameran kanssa onnistui. REMOTE-tila päällä yhteys ei toiminut. (DVT-Motoman Driver, [Viitattu 2.4.2012].)

# 6 DVT FRAMEWORK -OHJELMISTON KÄYTTÄMINEN




KUVIO 11. DVT FrameWorkin pääikkuna.

## 6.1 SoftSensors

SoftSensorit määrittelevät minkälaista tarkastelua SmartImage Sensor tekee. Jokaiselle SoftSensorille on määritelty yksilöllinen tehtävä tiedon etsimiseksi kame-

ralla otetusta kuvasta ja kaikkien käytettyjen tehtävien kombinaatio määrittelee millaista tarkastelua kamera tekee.

Result Table: Paikka

Overall Result FAIL

| SoftSensor | Result | < | Output Value |
|---|---|---|---|
| Paikka | PASS | | # Objects=1, X=221, Y=319, Angle=70 |
| coordSystem | PASS | | User Coordinate System |
| coordTransform | PASS | | X = -296.85, Y = 400.59 |
| Script | PASS | | x=-299, y=399, theta=(16384,0) a=-77.25 |
| C1 | FAIL | | Radius = 0.00 Pixels |
| C2 | FAIL | | Radius = 0.00 Pixels |
| C3 | FAIL | | Radius = 0.00 Pixels |
| C4 | FAIL | | Radius = 0.00 Pixels |

KUVIO 12. SoftSensorit ja tulokset.

### 6.2 Kappaleen paikoitus

Kappaleen paikoitus suoritettiin ObjectFind SoftSensorin avulla. ObjectFind SoftSensor luotiin valitsemalla se SoftSensorien listalta ja piirtämällä alue, jonka sisältä kappaletta etsitään.

SoftSensorille täytyi ensin opettaa etsittävän kappaleen muoto. Tämä tapahtui laittamalla työkappale yksin etsintäalueen sisälle ja painamalla ObjectFind SoftSensorin asetuksista Relearn Object -nappia.

Kameran valotusaikaa ja ObjectFind SoftSensorin asetuksia muutettiin kokeilemalla eri arvoja, kunnes ohjelma tunnisti kappaleen riittävän hyvin etsintäalueen eri kohdista.

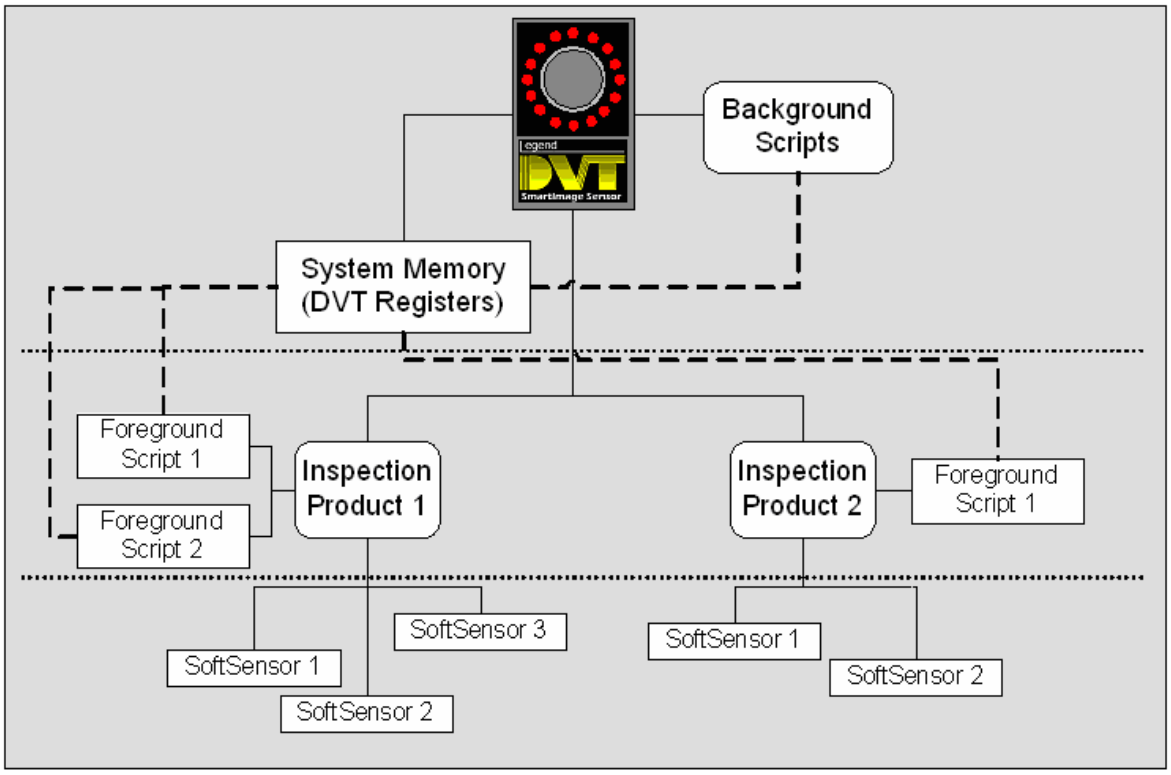
### 6.3 Skriptien käyttö

Vaikka SmartImage Sensorille on lukuisia eri SoftSensoreita suorittamaan erilaisia tehtäviä, eivät ne yleensä kuitenkaan riitä sovelluksen tarpeisiin ja näin ollen tarvitaan lisää muokkausmenetelmiä. Tähän tarkoitukseen on olemassa ohjelmoitavat

Script-työkalut, joiden toiminta ei ole ennalta määrättyä kuten SoftSensoreiden, vaan käyttäjä voi kirjoittaa niihin itse koodia sovelluksen tarpeiden mukaan. (DVT 2003.)

DVT FrameWorkissa skriptejä käytetään yleensä apuvälineenä käsittelemään SoftSensoreilta saatuja tietoja ja tuloksia. Käsittelyyn kuuluu yleensä laskutoimitusten tekeminen, tulosten tallentaminen DVT-rekistereihin ja lukeminen rekistereistä, sekä tulojen ja lähtöjen tarkkailu. (DVT 2003.)

FrameWork-ohjelmassa käytetään kahta eri ohjelmointityyppiä Foreground ja Background Scriptiä. Ne toimivat SmartImage Sensorissa eri tavoilla, kuten järjestelmän toimintaa esittelevä hierarkiakuva esittää. (DVT 2003.)




KUVIO 13. Skriptien toiminta järjestelmässä. (DVT 2003.)

### 6.3.1 Foreground Script

Foreground Scriptit luodaan sovellustasolla ja ne ovat suoraan yhteydessä tarkasteluihin. Skripti käynnistyy joka kerta, kun sovellus jossa se sijaitsee suorittaa kuvan tarkastelun. Foreground Scriptit luodaan kuten SoftSensorit ja siksi niitä kutsutaan Script SoftSensoreiksi. Niiden yleisimpiin tehtäviin kuulu tiedon kerääminen muista SoftSensoreista, laskelmien suorittaminen ja tiedon jakaminen muiden alueiden kanssa käyttäen järjestelmärekistereitä. Foreground Scriptejä voidaan myös käyttää tuomaan näkyville tarkastelun tuloksia, tarkastelemaan sisään- ja ulostuloja ja kirjoittamaan järjestelmämuistiin ja lukemaan sieltä. (DVT 2003.)

```
class Script
{
    public void inspect()
    {

        float x,y,a;

        x = coordTransform.TransformedPoint.X-3;
        y = coordTransform.TransformedPoint.Y-1;
        a = Paikka.ObjectAngle[0];

        a = a-2*a-96.80;

        if (a < -135)
        {
        a = a+90;
        }

        if (a > -45)
        {
        a = a-90;
        }

        DebugPrint(""+x);

        this.Position.X = x;
        this.Position.Y = y;
        this.Angle = a;

        RegisterWriteFloat(100,x);  //Kirjoitetaan paikkatiedot
        RegisterWriteFloat(200,y);  //DVT:n rekistereihin
        RegisterWriteFloat(300,a);
    }
}
```

KUVIO 14. Foreground Script.

Työhön luotiin yksi Foreground Scripti, joka keräsi paikkatiedot kappaleenpaikoitus SoftSensorilta ja kirjoitti ne DVT-kameran rekistereihin. Lisäksi skripti sisälsi laskutoimituksia, joilla saatiin paikkatiedot paremmin kohdilleen robotin koordinaatiston kanssa ja työkappaleen kiertokulma vastaamaan robotin työkalun kiertokulmaa.

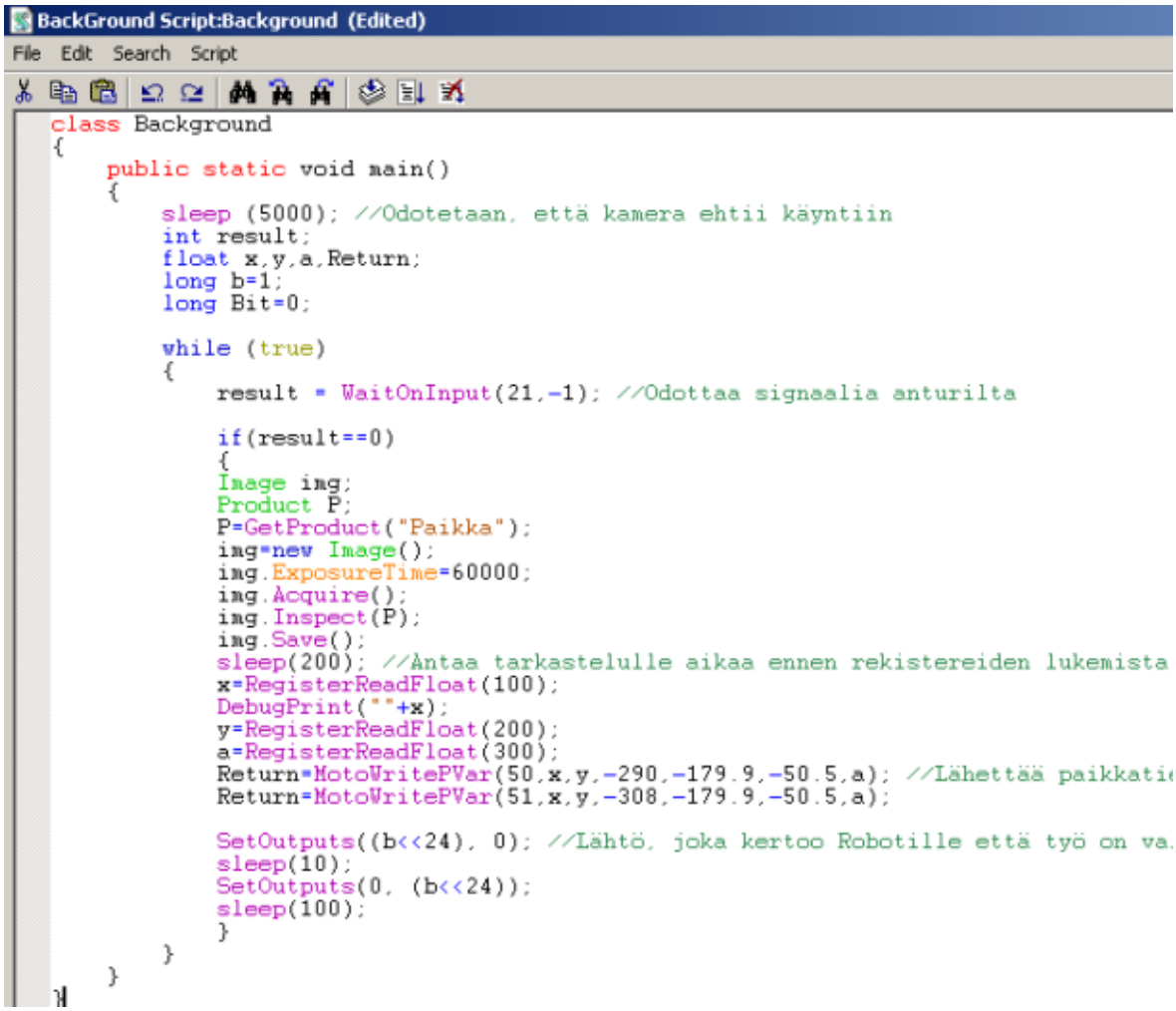
### 6.3.2 Background Script

Background Scriptin tehtäviä ovat mm. tarkasteltavan tuotteen valinta, tarkastelun käynnistäminen, rekistereiden lukeminen ja niihin kirjoittaminen, valotusajan säätäminen ja kommunikoiminen muiden laitteiden kanssa. Background Script toimii muiden ohjelmien taustalla, eikä sen suorittaminen ole kappaleiden tarkastelusta riippuvainen. Skriptin suorittamiseen on kolme tapaa:

- Käynnistys kameran päälle kytkennän yhteydessä: yleisesti käytetty, kun skriptin käsittelemät tiedot muuttuvat paljon ja skriptin halutaan olevan jatkuvasti päällä taustalla.

- Manuaalinen käynnistys: antaa käyttäjän luoda yhteyden SmartImage Sensoriin ja käynnistää Background Scriptin manuaalisesti. Käytetään yleensä luodun ohjelman testaamiseen.

- Käynnistys signaalista: käynnistyy kameran päälle kytkennän yhteydessä tai manuaalisesti, mutta jää odottamaan määrätyn tulon arvon muuttumista ennen skriptin suorittamista loppuun. (DVT 2003.)




KUVIO 15. Background Script.

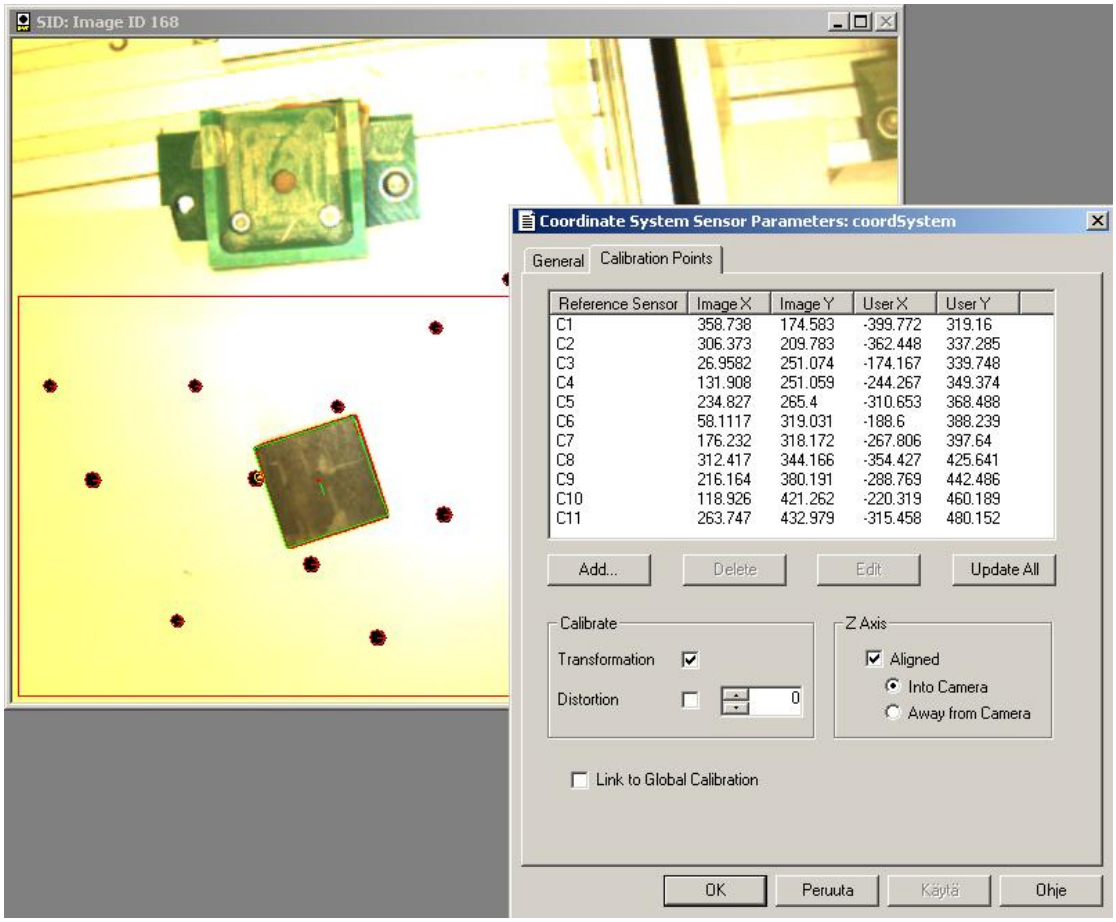
Tässä työssä Background Scriptin käynnisti robotilta tuleva signaali, jonka Motoman-robotti antoi kun se oli valmis vastaanottamaan kameralta paikkatietoa poimittavasta kappaleesta. Signaalin saatuaan skripti käski kameraa ottamaan kuvan ja suoritti tarkastelun. Tarkastelun yhteydessä Foreground Script keräsi paikkatiedot kuvasta, jotka Background Script luki DVT:n rekistereistä. Saadut tiedot tallennettiin Motomanin paikkamuuttujiin.

### 6.4 Kalibrointi robotin käyttämään koordinaatistoon

Konenäkökameran koordinaatisto täytyi muuttaa samaksi robotin käyttämän koordinaatiston kanssa, että robotille saatiin oikeat paikkatiedot kappaleesta. Tämä

onnistui FrameWorkin Coordinate Transformation ja Coordinate System SoftSensoreiden avulla.

Kameran kuva-alueelle asetettiin paperi, johon oli satunnaisiin kohtiin piirretty pisteitä. Näiden kalibrointipisteiden avulla Coordinate System SoftSensor pystyi muuttamaan FrameWorkin koordinaatit robotin koordinaatistoon sopiviksi. Ensin tarvittiin pisteiden nykyiset koordinaatit, jotka etsittiin FrameWorkissa kameran ottamasta kuvasta FindCircle SoftSensorin avulla. Tämän jälkeen tarvittiin vielä pisteille uudet koordinaatit, jotka saatiin robotilta ajamalla robotin tarttuja vuorotellen jokaisen pisteen kohdalle.




KUVIO 16. Pistepaperi ja koordinaatiston kalibrointi.

Coordinate Transformation SoftSensorilla liitettiin kappaleen paikan määrittävä Softsensori käyttämään uutta luotua koordinaatistoa. (DVT SmartImage Sensor Installation & User Guide, 116.)

# 7 ROBOTIN OHJELMA

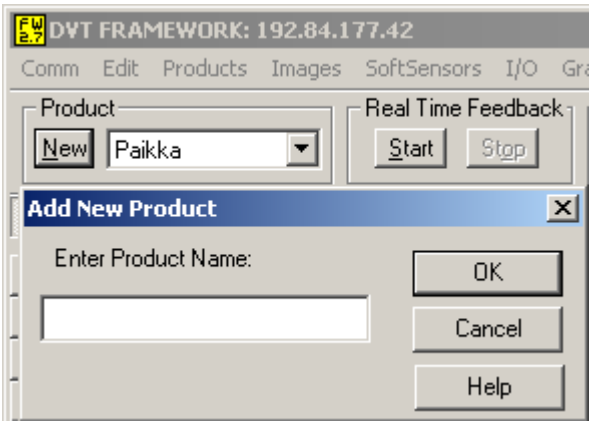
Robotin ohjelmointi tapahtui Motoman XRC MINI:n ohjelmointipaneelin avulla. Robotille luotiin yksinkertainen ohjelma, jossa se kameralta saatujen paikkatietojen avulla käy poimimassa työalueelta määrätynlaisen kappaleen ja kuljettaa sekä jättää sen valittuun paikkaan.

Ohjelman alussa robotti liikkuu aloitusasemaan ja lähettää kameralle signaalin PULSE OT#(8) -komennolla, joka ilmoittaa että robotti on valmis vastaanottamaan kappaleen paikkatiedot. Paikkatiedot luetaan kameralta SET B000 50- ja SAVEV B000 -komennoilla ja ladataan robotin paikkamuuttujaan LOADV P050 -komennolla. Tämän jälkeen robotti liikkuu paikkamuuttujan tietojen mukaan kappaleen kohdalle ja poimii sen. Lopuksi työkappale käydään jättämässä lopetuspaikkaan, jonka jälkeen robotti liikkuu takaisin aloitusasemiin.

# 8 KONENÄKÖSOVELLUKSEN LUOMINEN

Tässä sovelluksessa DVT:n konenäkökameralla tunnistettiin ja paikannettiin kappale robotin käsittelyä varten. DVT-kamera määritti kappaleen keskipisteen koordinaatit, tallensi tiedot rekistereihin, sekä lähetti paikkatiedot Motoman-robotille. Koordinaattien avulla robotti kävi poimimassa kappaleen ja kuljetti sen haluttuun päätepisteeseen.

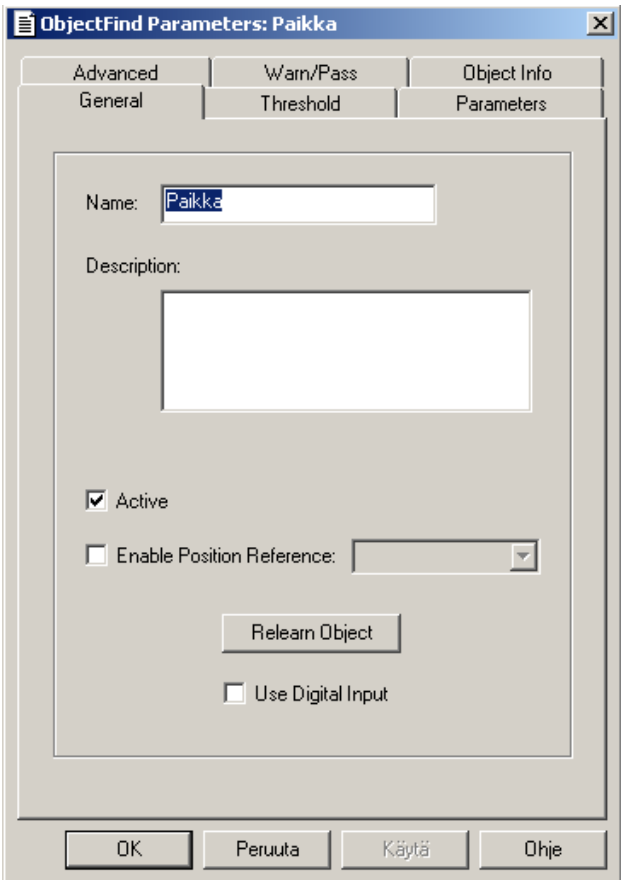
Sovelluksen luominen aloitettiin käynnistämällä konenäkökameran ohjelmisto DVT FrameWork 2.7.3 ja otettiin yhteys kameraan. Tämän jälkeen painettiin New Product -nappia ja luotiin uusi tuote "Paikka", johon tallentuivat kaikki sovelluksessa käytettävät SoftSensorit.




KUVIO 17. Uuden tuotteen luominen.

Kameran alle asetettiin kuvattava kappale ja säädettiin valaistus ja tarkennus sopiviksi. Lisäksi kuvausalue peitettiin valkoisella paperilla, joka oli hyvä tausta kuvattaessa mustaa kappaletta. Kuvan ollessa riittävän tarkka luotiin sensori, joka hoiti kuvattavan kohteen paikannuksen. Tähän tapaukseen sopi ObjectFind SoftSensor. Sensori luotiin piirtämällä suorakulmion muotoinen alue, jolta tarkastelua suoritettiin. Tämän jälkeen opetettiin sensorille paikoitettava kappale sijoittamalla se muuten tyhjälle kuvausalueelle ja painamalla sensorin asetuksista Relearn Object -nappia. Tämän jälkeen sensori osasi antaa kappaleen keskipisteen koor-

dinaatit ja kiertokulman. Kiertokulman nollakohta määrittyi Relearn Object -nappia painettaessa kappaleen senhetkisen asennon mukaan.

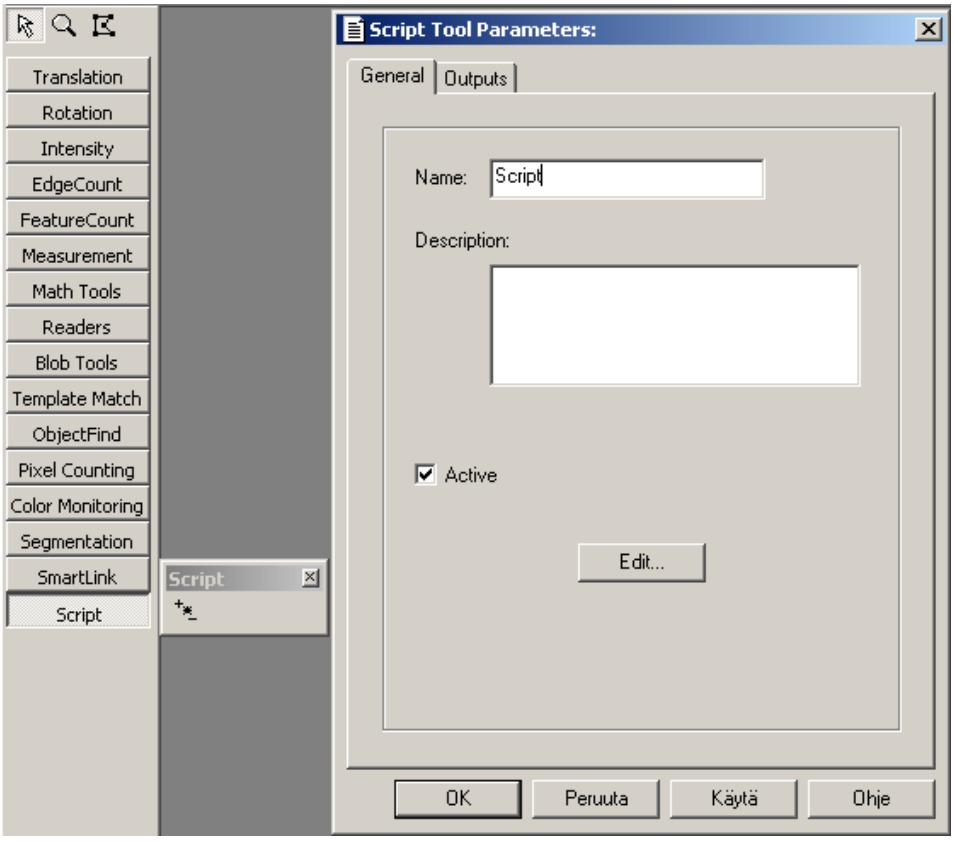



KUVIO 18. ObjectFind SoftSensor.

Seuraavaksi muutettiin FrameWork-ohjelman koordinaatisto vastaamaan robotin koordinaatistoa. Tähän käytettiin apuna pistepaperia, joka asetettiin kameran kuva-alueelle. Ensin ajettiin robotin tarttujaa pisteiden kohdille, jolloin saatiin robotin koordinaatit halutuista kohdista ja FrameWorkin vastaavat koordinaatit saatiin piirtämällä FindCircle SoftSensorit pisteiden kohdille. Tämän jälkeen luotiin Coordinate System -sensori, jonka asetusten Calibration Points -välilehdellä pystyttiin taulukkoon lisäämään FindCircle-sensorit ja niiden antamat koordinaatit. Samalla täytyi antaa manuaalisesti jokaista koordinaattia vastaavat uudet koordinaatit, jotka olivat tässä tapauksessa robottia ajamalla saadut paikkatiedot. Näin koordinaatiston muutos oli valmis, mutta se ei muuttanut koko ohjelman koordinaatistoa, vaan New Coordinate Transformation -sensorin avulla pystyi muuttamaan yksittäisen SoftSensorin koordinaatiston sen mukaan. Näin ollen luotiin uusi New Coor-

dinate Transformation -sensori ja sen asetuksista valittiin uudeksi koordinaatistoksi Coordinate System -sensori ja muutettavaksi sensoriksi ObjectFind-sensori.

Tämän jälkeen luotiin Foreground Script, jonka päätarkoitus oli ottaa ObjectFind-sensorin antamat paikkatiedot ylös ja tallentaa ne DVT-rekistereihin. SoftSensoreista valittiin Script, jonka jälkeen avautuvasta ikkunasta painettiin Edit-nappia jolloin päästiin kirjoittamaan skriptiä.




KUVIO 19. Foreground Scriptin luominen.

Ohjelmointi aloitettiin luomalla float-muuttujat x- ja y-koordinaatille sekä kulmalle. Seuraavaksi muuttujiin tallennettiin SoftSensoreilta saadut paikkatiedot. Kiertokulman arvo saatiin suoraan ObjectFind-sensorin kautta, mutta koordinaatit täytyi hakea New Coordinate Transformation -sensorin kautta, jolloin ne toimivat robotin kanssa. Lopuksi skriptissä kirjoitettiin float-muuttujien arvot rekistereihin. Foreground Script sisälsi myös laskutoimituksen, jolla kiertokulma käännettiin kiertämään samaan suuntaan kuin robotin työkalun kiertokulma. Lisäksi kahdella if-lauseella varmistettiin, ettei robotin tarvitse tehdä yli 90 asteen kiertymäliikettä,

vaikka sensorin antama kiertymä olisi sitä suurempi. Tämä oli mahdollista, koska kappale oli neliön muotoinen ja sillä oli neljä samanlaista sivua, josta robotti pystyi tarttumaan.

KUVIO 20. Sovelluksessa käytetty kappale.

Foreground Scriptin jälkeen täytyi luoda Background Script, jonka tehtävänä oli mm. hoitaa kommunikointi Motoman-robotin kanssa. Skriptin ohjelmointi aloitettiin jälleen luomalla muuttujat, joiden täytyi olla samaa tyyppiä kuin Foreground Scriptissä luotujen. Muuttujien määrittelyn jälkeen tuli while-silmukka, jonka sisällä olevaa koodia ohjelma suoritti Background Scriptin ollessa aktiivisena. Silmukan ulkopuolelle kirjoitettiin koodi, joka suoritettiin vain kertaalleen kameran käynnistyksen yhteydessä. While-silmukan sisään tehtiin if-lause, joka tarkkaili robotilta tulevaa tuloa ja käynnistyi pulssin saatuaan. Suurin osa Background Scriptin koodista tuli tämän if-lausekkeen sisään. Tähän koodiin kuului käytettävän tuotteen määrittäminen, tarkastelun käynnistäminen ja kuvan ottaminen, sekä Foreground Scriptissä käytetyistä rekistereistä lukeminen ja luettujen tietojen kirjoittaminen Motomanin paikkamuuttujiin. Skriptissä käytettiin myös muutaman kerran sleep-komentoa, joka antoi aikaa tehtävien suorittamiselle ennen seuraavien tehtävien käynnistymistä.

Skriptien ohjelmoimisessa käytettiin ohjelmointikieltä, joka oli sekoitusta Java, C- ja C++-ohjelmointikielistä. Valmiit ohjelmat näyttivät kuvioiden 14 ja 15 mukaisilta.

Lopuksi tehtiin ohjelma robotille, joka aloitettiin liikuttamalla robotti haluttuun aloituspaikkaan. Aloituspaikkaan saavuttuaan robotti lähetti ulostuloa pitkin pulssin, joka kertoi kameralle robotin olevan valmis kuvan ottamista ja paikkatietojen vastaanottamista varten. SET B000 50 määritti robotin B000-muuttujaan arvon, joka osoitti minkä paikkamuuttujan tiedot haluttiin DVT-järjestelmästä ladata ja LOADV P050 siirsi ne robotin P050-paikkamuuttujaan. Sama tehtiin P051-muuttujalle. Advinit-käsky esti samanaikaisen liikenteen. Seuraavaksi liikutettiin robottia saatujen paikkatietojen avulla ensin hieman kappaleen yläpuolelle, josta hoidettiin lähestyminen kappaleen kohdalle. Tätä seurasi tarttumiskomennot ja kappaleen liikkuminen paikkaan johon kappale jätettiin. Lopuksi liikutettiin robotti takaisin aloituspisteeseen, jolloin se oli valmiina suorittamaan ohjelman tarvittaessa uudelleen.

# 9 YHTEENVETO

Työn aihe vaikutti mielenkiintoiselta, sillä konenäön liittäminen robotin ohjaukseen loi monia erilaisia mahdollisuuksia ja automatisoi robotin toimintaa.

Konenäkökamera ja robotti olivat olleet jo ennestään liitoksissa toisiinsa, mutta tätä työtä aloittaessa niiden välinen yhteys ei toiminut. Laitteiden liitännät täytyi tarkastaa ja yhdistämisasetukset täytyi määrittää uudelleen. Yhteyden muodostamisessa esiintyi kuitenkin ongelmia ja vian määrittäminen oli hankalaa. Ensin konenäkökameraan ei saatu yhteyttä tietokoneelta, koska käyttäjätunnuksilla ei ollut riittäviä käyttöoikeuksia. Myöskään kameran ja robotin välinen yhteys ei aluksi toiminut ja syyksi paljastui viallinen kaapeli robotin ja sarjaporttipalvelimen välillä. Ongelmien takia työssä täytyi perehtyä huolellisesti myös yhdistämisasetuksiin ja laitteiden manuaaleihin.

Robotin ohjelman luomisessa rajoitteita asetti robotin tarttujatyökalu, jolla ei kovin erilaisia kappaleita pystynyt käsittelemään. Lisäksi rajoittavana tekijänä oli rajallinen tila, joka rajoitti robotin liikkumista. Tavoitteena ollut sovellus saatiin kuitenkin luotua ja aihe opetti uusia asioita teollisuusroboteista ja konenäöstä.

Aihetta olisi voinut jatkaa luomalla lisää esimerkkisovelluksia. Sovellukset olisivat voineet esitellä muita konenäön käyttömahdollisuuksia, kuten esimerkiksi värien ja muotojen tunnistamista. Näiden toteuttamiseen ei kuitenkaan jäänyt enää aikaa.

# LÄHTEET


Cognex. 2003. DVT SmartImage Sensor Installation & User Guide. [pdf-julkaisu]. Cognex Corporation. [Viitattu 29.2.2012]. Saatavissa: http://www.cognex.com/Support/Downloads/Download.aspx?d=818&l=2057.

DVT. 2004. What is “Machine Vision”?. [pdf-julkaisu]. DVT Corporation. [Viitattu 29.2.2012]. Saatavissa: http://automation.beijer.fi/web/webbfiles.nsf/0/46BBD1748B565A77C1256FD5004BD307/$File/WhatIsMachineVision.pdf.

Kuivanen, R. 1999. Robotiikka. Helsinki: Talentum Oyj.

Moxa. 2011. Nport 5200 Series User’s Manual. [pdf-julkaisu]. Moxa Inc. [Viitattu 19.3.2012]. Saatavissa: http://www.moxa.com/doc/man/NPort_5200_Series_Users_Manual_v7.pdf.

DVT-Motoman Driver. Ei päiväystä. [pdf-julkaisu]. DVT Corporation. [Viitattu 2.4.2012]. Saatavissa: http://www.sigavision.com/Downloads/Manuals/MotomanXRCDVTLegend.pdf.

RobotWorx. Ei päiväystä. Motoman SV3X. [pdf-julkaisu]. [Viitattu 26.3.2012]. Saatavissa: http://www.robots.com/pdfs/sv3x_datasheet.pdf.

RobotWorx. Ei päiväystä. Robot Parts: Mechanisms and Kinematics. [WWW-dokumentti]. [Viitattu 26.3.2012]. Saatavissa: http://www.robots.com/education/mechanisms/15

DVT. 2003. DVT Script Reference Manual. [pdf-julkaisu]. DVT Corporation. [Viitattu 4.4.2012]. Saatavissa: http://www.wallawalla.edu/academics/departments/engineering/students/classes/engr480/docs/Vision/dvt_script.pdf.

Groover, M. 2008. Automation, Production Systems, and Computer-Integrated Manufacturing. Kolmas painos. New Jersey: Pearson Education Inc.

# LIITTEET

## LIITE 1: Robotin ohjelma

```
0000    NOP
0001    MOVJ
0002    PULSE OT#(8)
0003    SET B000 50
0004    SAVEV B000
0005    SWAIT
0006    LOADV P050
0007    SWAIT
0008    SET B001 51
0009    SAVEV B001
0010    SWAIT
0011    LOADV P051
0012    SWAIT
0013    ADVINIT
0014    MOVL P050 V=100.0
0015    MOLV P051 V=100.0
0016    TIMER T=1.00
0017    DOUT OT#(2) ON
0018    TIMER T=1.00
0019    DOUT OT#(2) OFF
0020    MOVJ
0021    MOVJ
0022    MOVL
0023    TIMER T=1.00
0024    DOUT OT#(1) ON
0025    TIMER T=1.00
0026    DOUT OT#(1) OFF
```

**LIITE 1 (2): Robotin ohjelma**

```
0027    MOVL
0028    MOVJ
0029    END
```

**LIITE 2: ForeGround Script**

```
class Script
{
   public void inspect()
   {

      float x,y,a;

                              x = coordTransform.TransformedPoint.X-3;
                              y = coordTransform.TransformedPoint.Y-1;
                              a = Paikka.ObjectAngle[0];

                              a = a-2*a-96.80;
                              if (a < -135)
                              {
                              a = a+90;
                              }
                              if (a > -45)
                              {
                              a = a-90;
                              }

                              this.Position.X = x;
                              this.Position.Y = y;
                              this.Angle = a;

                              RegisterWriteFloat(100,x);    //Otetaan paikkatiedot ylös
DVT:n rekistereihin
                              RegisterWriteFloat(200,y);
                              RegisterWriteFloat(300,a);
   }
}
```

**LIITE 3: BackGround Script**

```
class Background
{
   public static void main()
   {
            sleep (5000); //Odotetaan, että kamera ehtii käyntiin
            int result;
            float x,y,a,Return;
            long b=1;
            long Bit=0;

            while (true)
            {
                     result = WaitOnInput(21,-1); //Odottaa signaalia anturilta

                     if(result==0)
                     {
                     Image img;
                     Product P;
                     P=GetProduct("Paikka");
                     img=new Image();
                     img.ExposureTime=60000;
                     img.Acquire();
                     img.Inspect(P);
                     img.Save();
                     sleep(200); //Antaa tarkastelulle aikaa ennen rekisterei-
den lukemista
                     x=RegisterReadFloat(100);
                     DebugPrint(""+x);
                     y=RegisterReadFloat(200);
                     a=RegisterReadFloat(300);
                     Return=MotoWritePVar(50,x,y,-290,-179.9,-50.5,a);
//Lähettää paikkatiedot Motomanille
```

**LIITE 3 (2): BackGround Script**

```
                    Return=MotoWritePVar(51,x,y,-308,-179.9,-50.5,a);



                    SetOutputs((b<<24), 0); //Lähtö, joka kertoo Robotille että
työ on valmis
                    sleep(10);
                    SetOutputs(0, (b<<24));
                    sleep(100);
                    }
      }
   }
}
```